Making Machine Learning Algorithms Work in Practice

Carla E. Brodley* and Padhraic Smyth†

* School of Electrical Engineering
Purdue University
West Lafayette, IN 47906
Phone: (317) 494-0635
FAX: (317) 494-6410

† Jet Propulsion Laboratory MS 525-3660
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109
Phone: (818) 306-6122
FAX: (818) 306-6912

brodley@ecn.purdue.edu, pjs@galway.jpl.nasa.gov

May 2, 1995

Abstract

In this paper we present a view of the overall *process* of application development for real-world classification and regression problems. Specifically, we identify, categorize and discuss the various problem-specific factors that influence this process.

1 Introduction

This paper considers the process of training a model from data and the issues involved in solving real-world prediction problems. As referred to in this paper, *models* are considered to be either classification or regression models. *Application development* is the overall process of applying a particular model (from a family of candidate models) to a domain-specific real-world classification or regression *problem*. The paper characterizes the application-development process and identifies the primary factors which influence the process.

What are the factors which influence the selection of a model for a particular application? While predictive accuracy may well be the main criterion for certain classes of practical classification and regression problems such as optical character recognition and speech recognition, we shall see that there are numerous other factors which influence the selection of a model for particular real-world problems.

The purpose of this paper is to explore the process of application development beyond the traditional limits of predictive accuracy and understandability criteria, which tend to dominate the research literature. In particular, the paper identifies the many factors which in practice affect application development and organizes these factors in a meaningful manner. Whether the

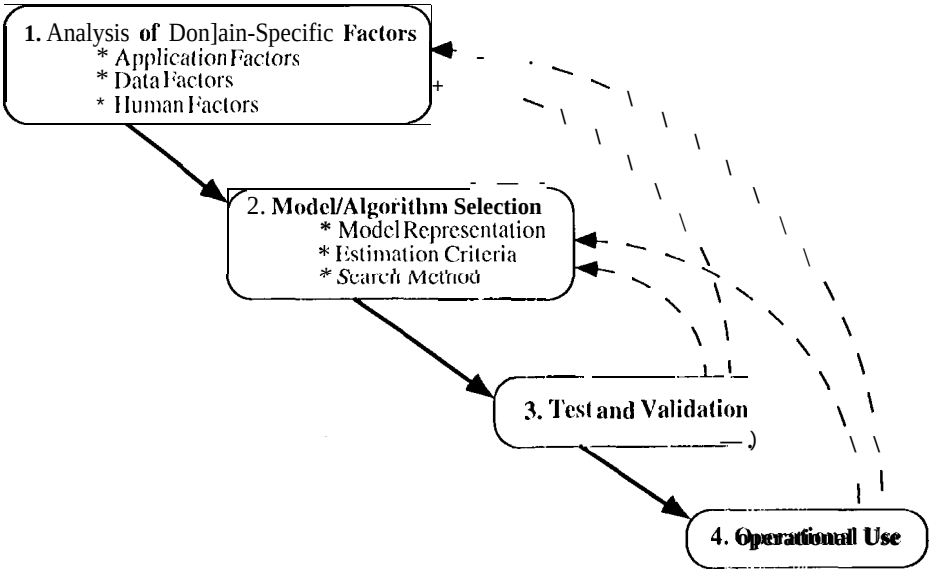

Figure 1: The application development process.

paper is successful or not in this endeavor must be judged by the reader: since these criteria are intrinsically *qualitative* in nature there is necessarily a subjective bias in the manner in which ideas are presented. The ideas as presented should assist practitioners in identifying salient criteria as they try to match general-purpose techniques and algorithms developed under research conditions with the particular vagaries of specific red-world classification and regression problems. in addition, the factors identified as part of the overall application development process should spur interest in the research community in the many practical issues which exist beyond the traditionally narrow scope of classical research in machine learning and pattern recognition.

The ideas presented have evolved from the authors' experiences with the practical application of classification) methods to real-world problems. As the various factors are identified and discussed, references will be made to particular applications (and published results whenever possible) where these issues have arisen.

The original motivation for developing a "big picture" of the application development process was partially inspired by the notions of meta-data analysis in statistical strategy modeling (Hand, 1993; Hand, 1994) where similar ideas have been explored in the context of understanding the overall process of fitting *statistical* models to data.

2 The Application Development Process

The process of developing a classification or regression application is typically iterative in practice and involves the consideration of various constraints and criteria. Figure 1 shows the four primary steps of this iterative process. The dotted arrows indicate that the process is iterative in nature. The four primary steps consist of:

1. Analysing the problem, which involves identifying the relevant domain factors, data factors, and human factors.

2. Selecting a particular model and algorithm based on matching the identified problem-specific factors (Step 1) to the general characteristics of the models and algorithms under considera-

tion

3. Analysis of the test results resulting from the selected model and algorithm. For most applications, a first stab at selecting an algorithm dots not lead to the final solution, but rather provides more data for re-analyzing the problem objectives and constraints. Indeed for many problems, many of the constraints can be altered, For example, one may be able to collect more data, which can change the ratio of problem dimensionality to sample size, which in turn influences the model and algorithm that one selects. Similarly, the analysis of the results of a particular algorithm can cause one to revise one's objectives. For example, although model understandability might initially have seemed like a primary goal, in the face of causing a significant loss of predictive accuracy it may become less important.

4. Finally, assuming the model satisfies the overall objectives of the project in a reasonable manner, the developed system is deployed in an operational environment.

In the next three sections we describe the steps of problem analysis and determination of relevant factors, model and algorithm selection based on these domain-specific factors, and iteration between analysis of test results and the first two steps before final operational use.

It is interesting to note that one can identify two common types of applications: (1) *generic* applications such as speech and optical character recognition (OCR) where different instances of the problem share many of the same characteristics, and (2) *specific* applications such as the classification of sky objects in astronomical images into the classes of star or galaxy (Fayyad, Smyth, Weir & Djorgovski, 1995). For generic applications, there is a continuous iterative process of application development which may last decades (as in speech) and techniques and results are shared by many groups working on the same problem. In contrast, specific applications tend to have a much shorter time-frame of development (on the order of 2 years or less) and the resulting techniques tend not to be as transferable.

3 Analysis and Identification of Problem-Specific Factors

The first step in the application development process is to *analyze* the problem and *identify* factors which are relevant to model and algorithm selection. In later sections we discuss the process of *matching* these domain-dependent factors to general domain-independent model and algorithm characteristics.

At a high level one can identify three primary *problem-dependent* factors which affect how a model and algorithm are chosen. For each primary factor one can identify various factors and criteria which are of relevance. These factors all into two groups: those that can be altered (such as acquiring more data) and those that cannot (such as the constraint that the classifier must be embedded within a larger system and must produce estimates of class probabilities given the feature data). The three primary problem-specific factors are:

1. **Domain Criteria and Factors:**
   This includes the overall objectives of the project, the amount of domain knowledge available about the problem, and operational factors which dictate how the model will be used in practice.

2. **Data Factors:**
   One of the significant differences between academic research and practical applications is in terms of how the data is collected. The researcher typically says "just send me a file of

labeled feature vectors on a floppy disk and 1'11 apply algorithm X ," whereas in practice the IIodel-builder is actively involved in the definition, acquisition and labeling of feature data,

3. **Human Factors:**
Who is the model-builder? Who is the customer/user? Who is the domain expert? Arc these the same or different people? In academic research the researcher "plays" all these roles in an artificial environment. In real applications there may be many different people involved with different levels of knowledge about the problem and the techniques being used.

In the remainder of this section wc discuss in more detail the factors for each component providing pointers to real-world applications where appropriate.

3.1 Domain Criteria and Factors:

1. **Objectives:**
What is the purpose of the application? Is the objective to test and field an operational model with high predictive accuracy? Is it a proof-of-concept? Or a statistical study of the potential effectiveness of competing models? 1 'requently the objectives are not made explicit at the start of a project which can compromise the whole application development process- this same problem is discussed in the context of statistical analysis by (Hand, 1994).

2. **Domain (Prior) Knowledge:**
Are the features for the problem well-defined or is feature extraction part of the problem? (Frequently in image analysis it is a major part of the overall classification problem (Fayyad, Smyth, Weir & Djorgovski, 1995)) What is known about the features (attributes) and classes individually? What is known about their relationships? Arc they causal or correlational? Is there a well-defined distance metric between the features? What is known about the sampling process? About the measurement p rocess? Can the prior knowledge be expressed in precise quantitative forms such as prior densities and distributions? Is there a single domain expert? If there arc multiple experts how well do they agree?

3. **Operational Factors:**
I1ow will the model be used in practice? Will training data be available online for updating the model? Does this consist of labeled or unlabeled data? Will the model require retraining by operational personnel? Are there computational requirements for speed and accuracy? (For example, JPL is currently considering the usc of image recognition algorithms for usc on robotic spacecraft visiting planets of the solar system: clearly there are severe restrictions 011 the computational demands of any such algorithms). Will there just be a single model produced or will there be many thousands (in which case average model performance may be more relevant than worsi-case)? What arc the actual loss functions involved? Is there a known loss matrix for classification decisions? Is it stationary over time? Is the model to be integrated into an overall system or is it a stand-alone application?

3.2 Data Factors

For real applications, data is rarely collected and labeled in advance: sampling schemes, labeling, missing data, feature representation arc all items over which the practitioner can exert some control *but at some economic cost.* The primary data factors are as follow's:

1. **Data Representation:**
   How are the features (attributes) represented: in continuous, discrete, categorical, hierarchical form, or some mixture of these forms? What is the dimensionality (the number of features) of the problem? Is the number of features fixed or variable? What is the form of the class variable? Are the classes mutually exclusive, mutually exhaustive? In fault detection, classes may be neither exhaustive nor exclusive (Smyth, 1994b). How many classes are there? For example, in speech and OCR applications there can be thousands of classes.

2. **Data Sampling:**
   Is there missing data? For the feature values or the classes? Is the available data a true random sample from the underlying population? If not is the sampling method known (e.g., a fixed number of samples per class)? What is the size of the training set? Is there measurement noise or outliers in the features? If so can this noise be characterized? Are the class labels reliable'f If not can the uncertainty be quantified (for example, in remote sensing applications the image data are often subjectively labeled after measurement by a human expert -- for some applications this labeling process can be quite noisy as described in (Burl, Fayyad, Perona, Smyth & Burl, 1 994))? Is the amount of training data fixed in advance or at the discretion of the user? Can the underlying density function be considered stationary over time?

3.3 Human Factors

Human factors can be difficult to assess and require effort to identify properly. 'J'here are three different sets of participants in a typical application:

1. Model Builder(s)
2. Domain Expert(s)
3. End-User(s) or Customer(s).

Typically these three sets of participants consist of multiple different individuals. At the other extreme all three sets consist of one person (this is rare in real operational applications).

For each of these sets of individuals one must consider their knowledge of the specific problem, its factors, and of the general model/algorithm characteristics. A I so relevant are their preferences and biases, their experience, and the "culture" of the particular problem domain (such as the degree of statistical and mathematical rigor expected in system design and validation).

Model selection for real applications is often carried out by individuals who are not experts in machine learning or pattern recognition algorithms: as such, one criteria is that the method be one they can trust (and understand), which may lead to using methods that do not necessarily obtain the highest predictive accuracy. One such example is the MultiSpec system (Landgrebe & Biehl, 1994), which is a data analysis system intended for multispectral image data, such I as that from the Landsat series of Earth Observational Satellites. MultiSpec's classification] method is well-known and simple it uses maximum likelihood discriminant analysis. The authors of MultiSpec have purposely kept the choice of model classes in MultiSpec small due to the human issue of not wanting MultiSpec to be a black box. Their users (who are mostly geographers and climatologists) are not up to date in the latest pattern recognition/ML techniques and would be wary of relying on such.

4 Model and Algorithm Selection

In this section we first describe the general *domain-indepen dent* characteristics that define a "learning algorithm." We then discuss how interactions between the factors presented in the previous section can influence the model and algorithm selection process. The process of model selection involves matching the domain-dependent factors described in the previous section to the domain-independent characteristics described Mow.

4.1 General Characteristics of Learning Algorithms

One of the more confusing aspects of learning algorithms is that there is such a variety of different algorithms published in the literature- when teaching machine learning or pattern recognition it is obvious that the student is easily confused by the bewildering number of available algorithms and methods. A useful "reductionist" view is the following: every learning algorithm can be viewed as consisting of the specification of the following three components:

1. **Model Representation:**
   What is the functional form of the model(s) being used by the algorithm? i.e., if the model can be expressed as $y = f(x, \theta)$ where x is the input, O represents the model parameters, and y is the prediction of the model f given x and O, what are the representational properties of the functional form of f?

2. **Estimation Criteria:**
   Given a particular representation f, estimation tells us what criteria we will usc to evaluate how well a particular set of parameters O fit the data. It is important to note that representation and estimation arc separate characteristics of a learning method and can be treated relatively independently (Cheeseman, 1990).

3. **Search Method:**
   Finally, given both a representational form (or a set of such forms) and an estimation criterion, the search method is the algorithmic specification of how the parameters and functions] forms arc fit to the data.

This three-component characterization (Buntine & Smyth, 1994; Fayyad, Piatetsky-Shapiro & Smyth, in press) is useful for identifying the distinguishing problem-relevant characteristics of different learning algorithms. For example, univariate decision tree methods can be viewed as consisting of (1) hierarchical piecewise constant mappings (representation), (2) likelihood and cross-validation criteria for node and tree select ion (estimation), and (3) various greed y growing and pruning strategies (scar ch). Similarly, feed-forward neural networks consist of (1) nonlinear mappings (representation), (2) likelihood-based objective functions (estimation), and (3) greedy gradient descent methods for weight selection (search in parameter space). Thus, the primary differentiating characteristics between decision trees and neural networks lie in their respective representations (trees arc more understandable but networks possess more flexible functions] forms).

For each of the three primary characteristics there are numerous sub-characteristics. In *representation* one is interested in the form of data that the model can handle (continuous, discrete, categorical, or all of these), the explanatory power of the model, the function approximation capabilities of the model, the form of the output of the model (class labels, posterior class probabilities, etc.), and so forth.

Under *estimation* the typical characteristics include: the sensitivity (or robustness) of the estimation criterion for a particular model as a function of sample size and as a function of the

dimensionality of the problem; the underlying assumptions of the estimation criterion (probabilistic, logical, independent sampling, etc.); and whether the estimation method can be applied automatically or requires some input from an expert user (e.g., Bayesian estimation methods typically require greater care on the part of the algorithm user than maximum likelihood methods).

Search characteristics include: the basic search methodology (greedy, exhaustive, hill-climbing, etc); the size of the scare]) space under consideration; the complexity of the search, whether it is just parameter search or also involve s a search over model structures; the ease of use of the method (whether it requires manual supervision); and the time and m emory complexity of the search.

4.2 Model and Algorithm Selection Based on Problem-Specific Factors

The particular problem defines the specific domain, dat a and human factors that constrain the model selection process. In the typical application, the domain factors (goals, prior knowledge, and operational issues) and human factors of a problem are often relatively fixed constraints, whereas the model-builder often has some latitude in terms of trade-oils involving data factors. An additional constraint that impacts the success of the model selection step is the model builder's knowledge about the domain-in dependent characteristics of the available learning algorithms. For example, if it is known that many of the features describing the data may be irrelevant, then an algorithm which performs feature selection (pruning) will be appropriate. In order to select such an algorithm the user must know which of the available algorithms perform this task (either explicitly as in the case of a sequential backward elimination process for linear discriminant functions, or implicitly as in a decision tree),

To select an appropriate learning algorithm one must know the affect that different data set characteristics have on the success of the algorithm for meeting the application objectives (Brodley, to appear; Box, 1990; Lehmann, 1990; Linhart & Zucchini, 1986). For example, in the domain o f remote sensing, the cost of collecting labeled training instances is prohibitive. In addition, with the recent advent of the AVRIS Imaging Spectrometer (which produces as many as 200 spectra] bands), there is a dramatic problem of too few training samples in relation to the number of features. For such a problem the Illodel-builder must understand the consequences of using particular models and estimation methods in situations where there is relatively little data relative to the dimensionality of the feature space.

Often human factors provide the overriding constraints that drive the model and algorithm selection process. For example, human factors played a large role in solving the problem of banding in rotogravure printing. In this application process delays (due to banding) were mitigated using the control rules disc overed by decision tree induction (Evans & Fisher, 1994). For this problem, the primary objective was that the end result of a machine learning algorithm provide an *operational* set of rules; the rules were used to change the way that the printing press is run.

The model builder aims to select the algorithm that maximizes the objectives of the problem, given both the problem-dependent factors and the domain-independent characteristics of the available learning algorithms. Typically there is no one solution which simultaneously satisfies all the constraints and optimizes all of the objectives. A successful application is often one which trades-off the various competing constraints to arrive at a useful solution satisfying most of the objectives.

5 Process Iteration

In an ideal world, the final steps of the process (after identifying the domain-dependent factors and selecting a model and algorithm) are to test the model and then field the system in an operational environment. Frequently, however, in real world applications a first attempt at algorithm selection

dots not provide satisfactory test results. This leads to iterating the overall process where the test results will suggest ways in which the objectives and criteria of the project can be altered or relaxed.

Specific situations which frequently occur in practice include:

- The estimated predictive accuracy of the model may be too low. A variety of factors can be examined: relevant variables for the problem are not being measured, important prior knowledge is being ignored, a larger training set is needed, the dimensionality of the problem is too high, the selected models and algorithms are inappropriate. For example, in (Burl, Fayyad, Perona, Smyth & Burl, 1994) analysis of the errors being made by a classifier trained to detect small volcanos in images of Venus revealed that the class labels provided by the domain experts were quite subjective in nature and that there was considerable disagreement among experts. This led to a complete re-evaluation of the overall problem, including the nature of the collection of training data, the training of the models, and the use of receiver-operating characteristics for model evaluation in the absence of absolute ground truth.

- Test results can reveal systematic errors that can be modeled and accounted for in the modeling process. For example, an initial attempt at online fault classification) from time series of large antenna pointing systems ignored the temporal aspect of the problem: the initial results suggested that an improved model could be obtained by embedding the initial classification model within a time dependent model such as a hidden Markov model (Smyth, 1994a). The improved model was subsequently adapted for use.

- For s_{ollic} applications, the initial results are successful enough in terms of predictive accuracy that closer attention is paid to the possible operational deployment of the model, which in turn can uncover new constraints. For example, in the fault classification problem mentioned in the previous paragraph it was only after a model with sufficient predictive accuracy was demonstrated that the real operational factors were scrutinized: in particular, the necessity of being able to detect classes which were not present in the training data was identified (Smyth, 1994b). This lead to further iteration through the model selection step of the process.

- The application development process itself may be *designed* to be iterative. In the MultiSpec system described earlier, the user and the system interact to achieve the best classification of an image. The first step involves *data review* in order for the user to gain familiarity with the data set, at least in part by viewing the data in color IR image form. The user then identifies the set of classes to be discriminated (for a particular application, not all possible classes will be of interest). Next the specific features to be used in the analysis must be identified or calculated (this is typically done by the system, but can be altered by the user). Finally, an analysis of the classifier built from the hand-labeled training data is performed and the results are evaluated using both quantitative and qualitative means. The results may lead the user to change the Set of classes of interest, change the features to be used, or change the analysis technique. The system was specifically designed to be interactive, because the acceptable results are both application and user specific, in particular, the definition of which classes are of interest.

- The initial results may force the project participants to more carefully evaluate domain-dependent factors such as misclassification costs. For example, in computer vision, classification is often an intermediate processing step rather than a final goal. In building a focus of attention mechanism using color and texture for triggering object matching routines, the objective is to build a classifier that makes one-sided errors. This objective arises because

it is less "expensive" to classify Hon-object pixels as object than classifying object pixels as non-object; missing the object is far more costly than applying a matching algorithm to non-object regions. In a particular application, road-following, a misclassification cost matrix was initially specified and a cost-sensitive algorithm was applied to the problem (Draper, Brodley & Utgoff, 1994). The results were unsatisfactory, causing a revision of the misclassification cost matrix. This process was iterated several times before an optimal cost-matrix was arrived upon.

These arc just a sample of many possible examples illustrating the iteraive nature of the application development process. If the domain-specific factors and the learning algorithm characteristics could be modeled precisely and their interactions predicted in a quantitative manner, there would be no need for this interactive, iterative process: one could simply identify the optimal solution. However, because the effects of interacting domain factors and algorithm characteristics cannot be predicted, the practical approach is to explore the solution space in an experimental but informed manner.

6 Conclusion

Typically the research community can make the strongest statements about algorithmic and model characteristics, e.g., "under these assumptions, algorithm A will produce behavior Y." However, it is fair to say that very little consideration is given in the research literature to domain-specific factors including domain, data and human factors. While it is understandable that these issues arc not the focus of much attention given their qualitative aud imprecise nature, nonetheless it is unfortunate that this is the case since in practical applications it is often the data and human issues which ultimately dictate success or failure of a project rather than algorithmic and model issues.

In this paper we have characterized the overall *process* of application development for real-world classification and regression problems and identified, categorized, and discussed some of the various factors which influence this process. We see the following potential benefits from this work:

- specification of a common framework to allow application developers to communicate and identify issues involved in the application development process,
- provision of a "road-map" for potential application developers, alerting the model-builder to the many potentially important issues which exist beyond the idealized research environment, and
- increasing the level of awareness in the research community concerning the large variety of factors which affect application development in practical situations. It is hoped that by identifying these factors that researchers will be encouraged to take an interest in addressing some of the many practical issues described in this paper, in particular, going beyond the focus on the criterion of predictive accuracy which tends to dominate the research literature at present to the exclusion of most other issues,

Acknowledgments

1'S would like to acknowledge discussions with Wray Buntine and Usama Fayyad which contributed to the ideas presented here. Part of the research described in this paper was performed at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration and was supported in part by ARPA and ONR under grant number N00014-92-J-1860.

References

Box, D.R. (1990). Role of models in statistical analysis. *Statistical Science, 5,* 169-)74.

Brodley, C.E. (to appear). Recursive automatic bias selection for classifier construction. *Machine Learning.*

Buntine, W., & Smyth, 1'. (1994). Learning from data: A probabilistic framework. *Tutorial notes for AAAI-94 conference.* Menlo Park, CA: AAAI.

Burl, M.C., Fayyad, U.M., Perona, 1'., Smyth, 1'., & Burl, M. 1'. (1994). Automating the hunt for volcanoes on Venus. *Proceedings of the 1994 Computer Vision and Pattern Recognition Conference (CVPR-94)* (pp. 302-309). Los Alamitos, CA: IEEE Computer Society Press.

Cheeseman, P. (1990). On finding the most probable model. In Shrager & Langley (Eds.), *Computation Models of Scientific Discovery and Theory Formation.* San Mateo, CA: Morgan Kaufmann.

Draper, B. A., Brodley, C.E., & Utgoff, P.E. (19$)4), Goal-directed classification using linear machine decision trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 16,* 888-893.

Evans, B., & Fisher, D. (1994). Overcoming process delays with decision tree induction. *IEEE Expert,* 9,6066.

Fayyad, J. M., Smyth, P., Weir, N., & Djorgovski, S. (1995). Automated analysis and exploration of large image databases. *Journal of Intelligent Information Systems, 4,* 7-25.

Fayyad, U. hi., Piatetsky-Shapiro, G., & Smyth, 1'. (in press). From data-mining to knowledge discovery: An overview. In Fayyad, Piatetsky-Shapiro, Smyth & Uthurasamy (Eds.), *Advances in Knowledge Discovery and Data Mining.* AAAI/MIT Press.

Hand, D. J. (1993). *Artificial Intelligence Frontiers in Statistics: AI and Statistics III.* London, UK: Chapman and Hall.

Hand, D. J. (1994). Statistical strategy: Step 1. In Cheeseman & Oldford (Eds.), *Selecting Models from Data: Artificial Intelligence and Statistics IV.* New York: Springer-Verlag.

Landgrebe, D., & Biehl, L. (1994), *An Introduction to Multispec.* Purdue Research Foundation.

Lehmann, E. L. (1990). Model specification: The views of Fisher and Neyman, and later developments. *Statistical Science, 5,* 160-168.

Linhart, **11.,** & Zucchini, W. (1986). *Model Selection.* NY: Wiley.

Smyth, 1). (1994a). Hidden Markov monitoring for fault detection in dynamic systems. *Pattern Recognition,* 27, 149-164.

Smyth, P. (1994b). Markov monitoring with unknown states. *IEEE Journal on Selected Areas in Communications, special issue on intelligent signal processing for communications, 12,* 1600-1612.